THESIS

TRANSITIONING CLIENT-BASED NALCOMIS TO A MULTI-FUNCTION WEB-BASED APPLICATION

by

Aaron P. Schnetzler

September 2016

Thesis Co-Advisors: Man-Tak Shing
Arijit Das

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704–0188*

1. AGENCY USE ONLY *(Leave Blank)*	**2. REPORT DATE** 09-23-2016	**3. REPORT TYPE AND DATES COVERED** Master's Thesis 07-08-2013 to 09-23-2016

4. TITLE AND SUBTITLE TRANSITIONING CLIENT-BASED NALCOMIS TO A MULTI-FUNCTION WEB-BASED APPLICATION	**5. FUNDING NUMBERS**
6. AUTHOR(S) Aaron P. Schnetzler	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School, Monterey, CA 93943	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE**

13. ABSTRACT *(maximum 200 words)*

Navy and Marine Corps aviation utilizes a software suite to manage logistics known as NTCSS, and one of its primary applications is NALCOMIS. NALCOMIS has two configurations that are used by organizational and intermediate level maintenance activities, Optimized Organizational Maintenance Activity (OOMA) and Optimized Intermediate Maintenance Activity (OIMA). These configurations communicate with each other when co-located with a local client server. If a squadron departs its home station for a deployment or exercise, the OOMA server is disconnected from OIMA and lines of communication are lost. All data that needs to be shared between systems must be manually entered and updated. Manual data entry can lead to errors, resulting in inventory discrepancies that can amount to millions of dollars. This research examines technologies that lead to the design of a system that seamlessly integrates the two configurations of NALCOMIS and moves from a local client server model to a web server accessed through a secure web application. A proof of concept was developed to demonstrate the viability and utility of the proposed web-based application. Our analysis of the system load on the existing NALCOMIS servers shows that, with a minimum investment in hardware, a Marine Corps wide NALCOMIS-WEB could be implemented to create a fully interconnected Marine Aviation Logistics Squadron (MALS) network.

14. SUBJECT TERMS NALCOMIS, database, web server, JDBC, web application, multi-tier architecture, aviation logistics	**15. NUMBER OF PAGES** 61
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500 **Standard Form 298 (Rev. 2–89)** **Prescribed by ANSI Std. 239–18**

THIS PAGE INTENTIONALLY LEFT BLANK

TRANSITIONING CLIENT-BASED NALCOMIS TO A MULTI-FUNCTION WEB-BASED APPLICATION

Aaron P. Schnetzler
Major, United States Marine Corps
B.S., United States Naval Academy, 2003

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2016

Approved by: Man-Tak Shing
Thesis Co-Advisor

Arijit Das
Thesis Co-Advisor

Dr. Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Navy and Marine Corps aviation utilizes a software suite to manage logistics known as NTCSS, and one of its primary applications is NALCOMIS. NALCOMIS has two configurations that are used by organizational and intermediate level maintenance activities, Optimized Organizational Maintenance Activity (OOMA) and Optimized Intermediate Maintenance Activity (OIMA). These configurations communicate with each other when co-located with a local client server. If a squadron departs its home station for a deployment or exercise, the OOMA server is disconnected from OIMA and lines of communication are lost. All data that needs to be shared between systems must be manually entered and updated. Manual data entry can lead to errors, resulting in inventory discrepancies that can amount to millions of dollars. This research examines technologies that lead to the design of a system that seamlessly integrates the two configurations of NALCOMIS and moves from a local client server model to a web server accessed through a secure web application. A proof of concept was developed to demonstrate the viability and utility of the proposed web-based application. Our analysis of the system load on the existing NALCOMIS servers shows that, with a minimum investment in hardware, a Marine Corps wide NALCOMIS-WEB could be implemented to create a fully interconnected Marine Aviation Logistics Squadron (MALS) network.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

ACE	aviation combat element
APES	Automated Performance Evaluation System
API	application program interface
AVLOG	aviation logistics
CA	certificate authority
CAC	common access card
CE	command element
CSS	cascading style sheets
DBMS	database management software
DLA	Defense Logistics Agency
DOD	Department of Defense
DPS	Defense Personal Property System
DTS	Defense Travel System
EAF	expeditionary airfield
GCE	ground combat element
HTML	hypertext markup language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICC	integrated circuit chip

ICP	inventory control points
IMA	intermediate maintenance activity
IMPR	inventory management performance report
JAR	Java archive
JDBC	Java Database Connectivity
JDK	Java Development Kit
LCE	logistics combat element
MAG	Marine Aircraft Group
MAGTF	Marine air-ground task force
MALS	Marine Aviation Logistics Squadron
MARFORPAC	Marine Forces Pacific
MAW	Marine Aircraft Wing
MEB	Marine Expeditionary Brigade
MEF	Marine Expeditionary Force
MEU	Marine Expeditionary Unit
MOL	Marine Online
NAE	Naval Aviation Enterprise
NALCOMIS	Naval Aviation Logistics Command Management Information System
NAVSUP	Naval Supply Systems Command
NIIN	National Item Identification Number
NTCSS	Naval Tactical Command Support System
ODBC	open database connectivity

OIMA	Optimized Intermediate Maintenance Activity
OMA	organizational maintenance activity
OOMA	Optimized Organizational Maintenance Activity
PIN	personal identification number
PKI	public key infrastructure
RDBMS	relational database management software
RSUPPLY	Relational Supply
SQL	Structured Query Language
URL	uniform resource locator
VA	validation authority
WAR	web archive

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

I would like to thank my classmates and professors for all their help the last two years on this journey to a master's degree. "ryeGuy" always came through in the clutch.

Thanks to Dr. Shing and Mr. Das for their incredible patience and assistance during the thesis process.

Thank you CrossFit Monterey, TZ Strength, Top Tier Barbell, and @nutmeg_nonsense for providing an outlet for my frustrations and a place to talk about anything except computer science.

Lastly, there are not enough thanks that can go to my wife, Chie, and daughter, Miki, for putting up with an absentee husband/father while I've been parked in front of a computer trying to finish this.

Dōmo arigatōgozaimasu, watashi wa anata no ryōhō o aishiteimasu.

HCF XPR SDI

rm -rf /

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1: Introduction

1.1 Navy and Marine Aviation

The collective air power of United States Navy and Marine Corps aviation totals over 1600 aircraft stationed and deployed around the world. They consist of fixed-wing, rotor-wing, and unmanned assets that fill a variety of combat and support roles. Like any machine, these aircraft require maintenance and new parts to keep them performing at optimum levels. Naval aviation maintenance actions and supply requests are tracked in the program suite known as Naval Tactical Command Support System (NTCSS), the two main programs of which are Naval Aviation Logistics Command Management Information System (NALCOMIS) and Relational Supply (RSUPPLY). Both are database-heavy programs designed to track and update maintenance requests, actions, parts orders, inventories, etc.

1.2 NALCOMIS

Navy and Marine Corps aviation likes to have cutting-edge technology in the air, but its information systems sometimes lag behind. Currently, a solution does not exist to provide real-time updates to the existing NALCOMIS database when flying squadrons deploy from their home stations to participate in training exercises.

When deployed, maintenance and supply personnel often resort to hand-written methods to place new requisitions for materiel, and those paper copies are then manually entered at a later time into the NALCOMIS system.

1.3 Objectives

The main objective of this thesis is to utilize commercially available technologies to develop a proof of concept of an online version of NALCOMIS that can be accessed at anytime over the Internet.

Research areas included databases, web servers, and application program interfaces (APIs); and how to integrate the three into a functional web-based database suitable for aviation logistics, culminating in a proof of concept design for a functional web-based NALCOMIS.

1.4 Marine Corps Web Based Applications

This idea of transitioning a system to the Internet is not without precedent. More and more government systems and processes are moving to web-based applications that can be accessed from anywhere. Many of the Marine Corps's administrative tasks are available via Marine Online (MOL), and all fitness reports are now processed online via Automated Performance Evaluation System (APES). Department of Defense (DOD) wide applications such as Defense Travel System (DTS) and Defense Personal Property System (DPS) are now the norm instead of paper-based systems. Transitioning a system that is of vital importance to units that have high operational tempos to a web-based platform only makes sense, and is the next step in modernizing current systems.

1.5 Scope

Chapter 2 of this work will provide a background in Marine aviation and aviation logistics, and an introduction into the components of the NTCSS: RSUPPLY and NALCOMIS.

In Chapter 3, current platforms in web server, databases, and Java APIs are examined and how they can be used to create a functional web-based version of NALCOMIS, and addresses security concerns of DOD websites.

Chapter 4 will outline current usage of NALCOMIS, provide a proof of concept of a web-based NALCOMIS that could be used in a deployed environment, and examine a model of data throughput that could be used to develop hardware requirements for a fully web-based version of NALCOMIS.

Finally, Chapter 5 will summarize the work and discuss future areas to expand the proof of concept and other areas of research.

CHAPTER 2: Marine Aviation and Logistics Structure

2.1 USMC Aviation

The Marine Corps is America's expeditionary force in readiness, ready to defend the nation's interests in the air, on land, and sea. In order to accomplish this, Marine units are organized into various-sized elements that collectively make up the principle organization known as the Marine air-ground task force (MAGTF). The MAGTF itself is composed of the command element (CE), ground combat element (GCE), logistics combat element (LCE), and the aviation combat element (ACE); and the MAGTF can come in various sizes: the Marine Expeditionary Force (MEF), the Marine Expeditionary Brigade (MEB), and the Marine Expeditionary Unit (MEU).

The Marine Corps ACE is the air element of the MAGTF and provides fixed-wing, rotor-wing, tilt-rotor, and unmanned assets, with appropriate supporting units and personnel, to the MAGTF commander. "The majority of aircraft usage within the MAGTF is for close air support or transport for the GCE or LCE, however, other specialized missions are available. The six main functions include: assault support, anti-air warfare, offensive air support, electronic warfare, control of aircraft and missiles, and aerial reconnaissance" [1].

The ACE is dependent on the organization of the MAGTF; a MEF has a Marine Aircraft Wing (MAW); a MEB holds a Marine Aircraft Group (MAG); and a MEU has a reinforced squadron containing a mixture of aircraft [1]. The structure of the MEF and the MAW are outlined in Figures 2.1 and 2.2, respectively. Providing support to these units is the primary mission of Marine aviation logistics (AVLOG).

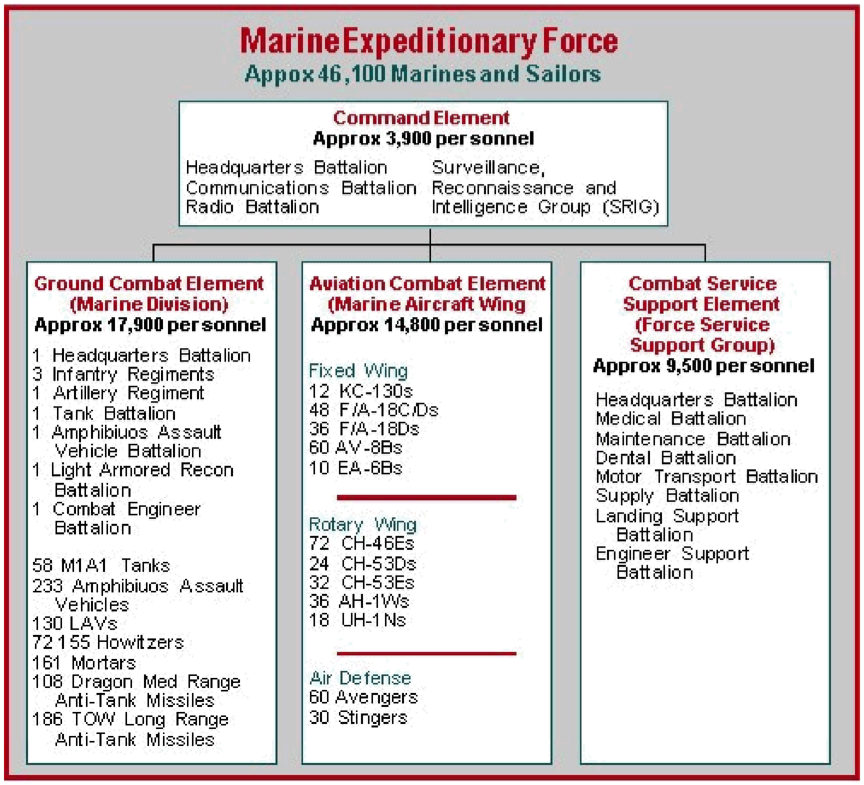


Figure 2.1. MEF Structure. Source: [2].

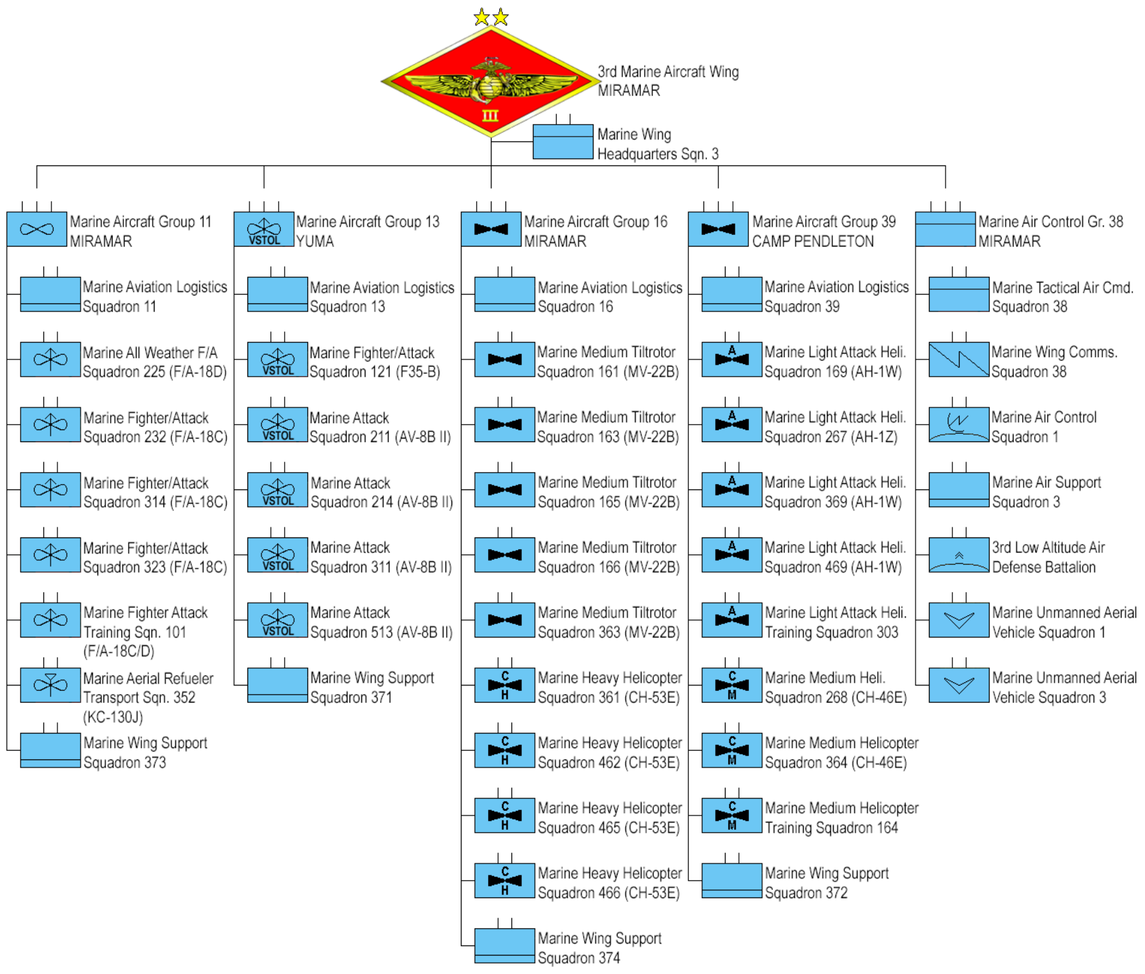

3rd Marine Aircraft Wing MIRAMAR
III
Marine Wing Headquarters Sqn. 3
Marine Aircraft Group 11 MIRAMAR
Marine Aviation Logistics Squadron 11
Marine All Weather F/A Squadron 225 (F/A-18D)
Marine Fighter/Attack Squadron 232 (F/A-18C)
Marine Fighter/Attack Squadron 314 (F/A-18C)
Marine Fighter/Attack Squadron 323 (F/A-18C)
Marine Fighter Attack Training Sqn. 101 (F/A-18C/D)
Marine Aerial Refueler Transport Sqn. 352 (KC-130J)
Marine Wing Support Squadron 373
Marine Aircraft Group 13 YUMA
VSTOL
Marine Aviation Logistics Squadron 13
Marine Fighter/Attack Squadron 121 (F35-B)
Marine Attack Squadron 211 (AV-8B II)
Marine Attack Squadron 214 (AV-8B II)
Marine Attack Squadron 311 (AV-8B II)
Marine Attack Squadron 513 (AV-8B II)
Marine Wing Support Squadron 371
Marine Aircraft Group 16 MIRAMAR
Marine Aviation Logistics Squadron 16
Marine Medium Tiltrotor Squadron 161 (MV-22B)
Marine Medium Tiltrotor Squadron 163 (MV-22B)
Marine Medium Tiltrotor Squadron 165 (MV-22B)
Marine Medium Tiltrotor Squadron 166 (MV-22B)
Marine Medium Tiltrotor Squadron 363 (MV-22B)
Marine Heavy Helicopter Squadron 361 (CH-53E)
Marine Heavy Helicopter Squadron 462 (CH-53E)
Marine Heavy Helicopter Squadron 465 (CH-53E)
Marine Heavy Helicopter Squadron 466 (CH-53E)
Marine Wing Support Squadron 374
Marine Aircraft Group 39 CAMP PENDLETON
Marine Aviation Logistics Squadron 39
Marine Light Attack Heli. Squadron 169 (AH-1W)
Marine Light Attack Heli. Squadron 267 (AH-1Z)
Marine Light Attack Heli. Squadron 369 (AH-1W)
Marine Light Attack Heli. Squadron 469 (AH-1W)
Marine Light Attack Heli. Training Squadron 303
Marine Medium Heli. Squadron 268 (CH-46E)
Marine Medium Helicopter Squadron 364 (CH-46E)
Marine Medium Helicopter Training Squadron 164
Marine Wing Support Squadron 372
Marine Air Control Gr. 38 MIRAMAR
Marine Tactical Air Cmd. Squadron 38
Marine Wing Comms. Squadron 38
Marine Air Control Squadron 1
Marine Air Support Squadron 3
3rd Low Altitude Air Defense Battalion
Marine Unmanned Aerial Vehicle Squadron 1
Marine Unmanned Aerial Vehicle Squadron 3

Figure 2.2. MAW Structure. Source: [3].

2.2 Marine Corps Aviation Logistics

Marine Corps aviation plays a key role within the entire Naval Aviation Enterprise (NAE). Commanders and aviation logisticians coordinate in order to conduct aviation-specific logistical operations and provide support to the ACE and the MAGTF all over the world.

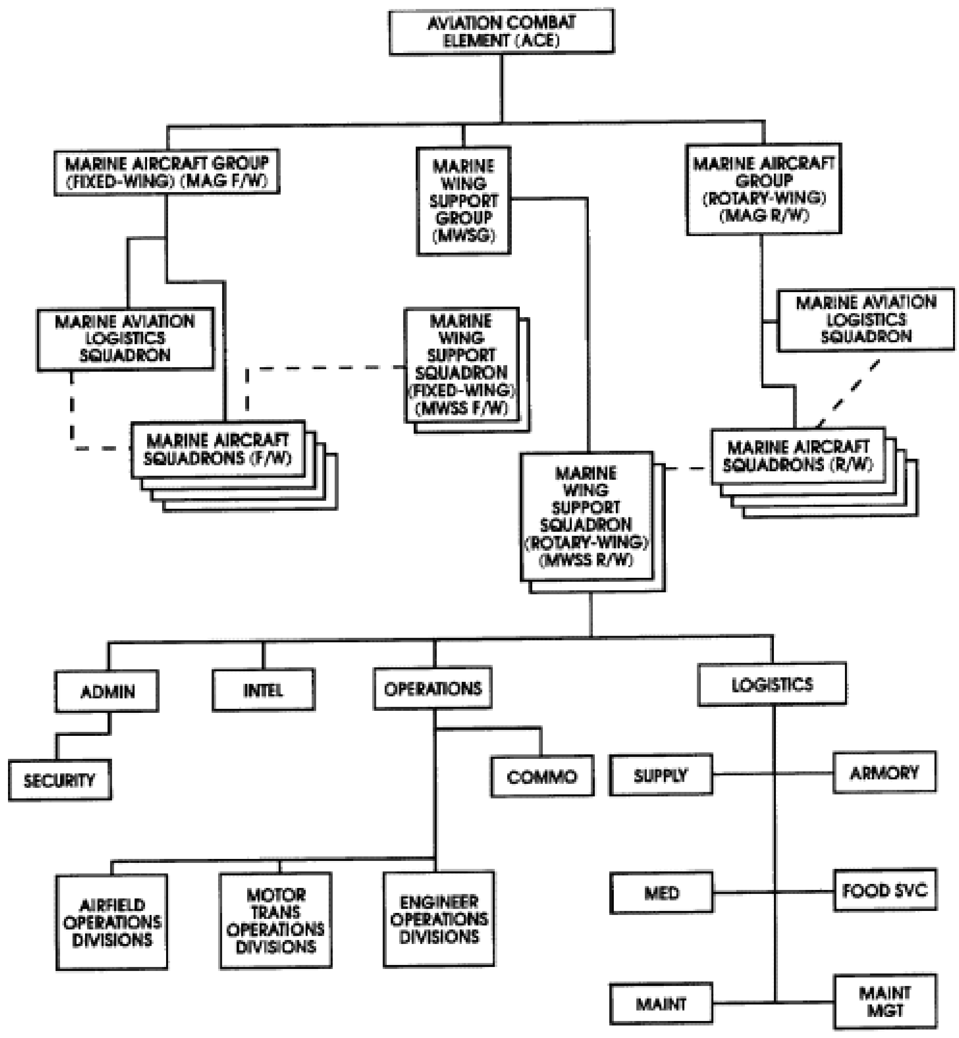
As stated in the Aviation Logistics Marine Corps Warfighting Publication (MCWP), "the AVLOG mission of the Marine Corps, at all command and support levels, is to assist in generating ACEs that are rapidly deployable, self-reliant, self-sustaining, and flexible:

- Rapid deployment demands that ACE organizations, equipment, and supplies be readily transportable by land, sea, and air.
- A self-reliant ACE is task-organized to support itself logistically with accompanying supplies for specific time frames without undue concern for resupply or developed infrastructure ashore (Figure 2.3).
- An ACE's AVLOG capabilities and accompanying supplies enable it to sustain its own operations for up to 90 days while external resupply channels are organized and established.
- An ACE's inherent self-sustainment and rapid deployability capabilities allow it to quickly reconstitute itself and permit rapid withdrawal from a completed operation and immediate re-embarkation for follow-on missions [4].

Marine AVLOG is supported by Naval Supply Systems Command (NAVSUP) and Defense Logistics Agency (DLA) to provide depot level repairable and consumable items. These two organizations have inventory control points (ICP) all over the world in order to provide pre-positioned items and shorten the amount of time to deliver to customers (Figure 2.4).

When replacement parts are required for aircraft, the flying squadron will order through the Marine Aviation Logistics Squadron (MALS). The MALS is the tactical level logistics unit for Marine AVLOG, and is the go-between for flying squadrons and NAVSUP and DLA.

The MALS is composed of several departments with multiple divisions within each. It functions as the intermediate maintenance activity (IMA), or I-level, and completes repairs that cannot be performed by the flying squadron's maintenance department; the organizational maintenance activity (OMA), or O-level.


AVIATION COMBAT ELEMENT (ACE)
MARINE AIRCRAFT GROUP (FIXED-WING) (MAG F/W)
MARINE WING SUPPORT GROUP (MWSG)
MARINE AIRCRAFT GROUP (ROTARY-WING) (MAG R/W)
MARINE AVIATION LOGISTICS SQUADRON
MARINE WING SUPPORT SQUADRON (FIXED-WING) (MWSS F/W)
MARINE AVIATION LOGISTICS SQUADRON
MARINE AIRCRAFT SQUADRONS (F/W)
MARINE WING SUPPORT SQUADRON (ROTARY-WING) (MWSS R/W)
MARINE AIRCRAFT SQUADRONS (R/W)
ADMIN
INTEL
OPERATIONS
LOGISTICS
SECURITY
COMMO
SUPPLY
ARMORY
AIRFIELD OPERATIONS DIVISIONS
MOTOR TRANS OPERATIONS DIVISIONS
ENGINEER OPERATIONS DIVISIONS
MED
FOOD SVC
MAINT
MAINT MGT

- - - - - Aviation Logistics Support. Consists of aviation supply, intermediate maintenance of aircraft, aviations armament, ordnance, and support equipment.

- - - - - Aviation Ground and Combat Service Support. Consists of airbases support functions and CSS functions for ground supply, maintenance, transportation, engineering, health services, and services.

Figure IX-4. Aviation Combat Element Logistics Organization

Figure 2.3. ACE Logistics. Source: [5].

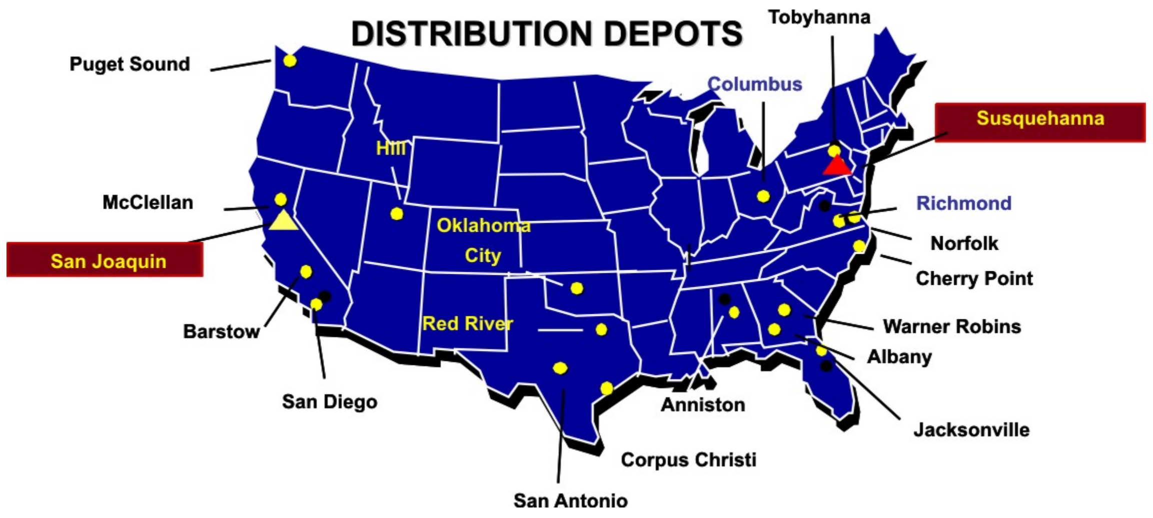


Figure 2.4. DLA CONUS Depots. Source: [6].

Within the MALS, the supply department is responsible for the stocking, warehousing, issuing, receipt, delivery, and expediting of parts used by the OMA. To accomplish this, both the I-Level and O-level utilize a software suite called NTCSS, which contains the applications RSUPPLY and NALCOMIS.

2.2.1 RSUPPLY

RSUPPLY provides three management tools to aviation logistics personnel in the areas of logistics, inventory, and financial management. The AVLOG MCWP provides simplified descriptions of the three subsystems.

The logistics subsystem enables a variety of materiel transactions including: "automated assistance for supply department materiel control and customer support activities; online collection and maintenance of data on stock items, repairables, and requisitions; online requesting of materiel by supply department customers and automated issue of materiel or creation of requisitions; and automation to manage offload or stock transfer" [4].

The inventory management subsystem primarily handles maintenance, computing, adjusting, and reporting inventory data. Maintenance of data involves establishing and maintaining records "that identify, locate, quantify, and describe stock items. Actual materiel on hand and materiel due versus materiel received are reconciled; and surveys, gains, or losses are processed" [4]. Another function "implements policies through system-wide inventory data modifications and produces management reports that summarizes stock item information", and allows personnel to set stocking objectives and allowances [4].

Finally, the MCWP describes the financial management subsystem to allow users "manual or automated updates and information queries of all financial data" in the system [4]. It is comprised of three functions: "providing automated support for maintaining up-to-date financial data, monitoring and controlling fund expenditures, and producing financial reports and displays." Financial records for all supply transactions are maintained in this subsystem; it also provides data to reports and queries, "and provides controls to promote accuracy and validity of financial data" [4].

2.2.2 NALCOMIS

NALCOMIS is designed in several configurations to work in both the O-level and I-level. The AVLOG MCWP states that it provides both activities a real-time information system designed to:

- Increase aircraft and aeronautical equipment readiness by providing local maintenance and supply managers with timely and accurate information.
- Reduce the administrative burden on the fleet.
- Improve the quality of upline reported data [4].

In its current setup (Figure 2.5), NALCOMIS operates in two configurations, Optimized Organizational Maintenance Activity (OOMA) and Optimized Intermediate Maintenance Activity (OIMA). A mid-tier server sits between the two and processes data back and forth between them. The three servers are all tied together when flying squadrons are operating out of their home station. When squadrons deploy, though, the connection is lost and alternate methods are used when parts are requisitioned; these methods may include fax, email, and/or telephone. These methods introduce variation into the data entry process and more opportunity for errors.

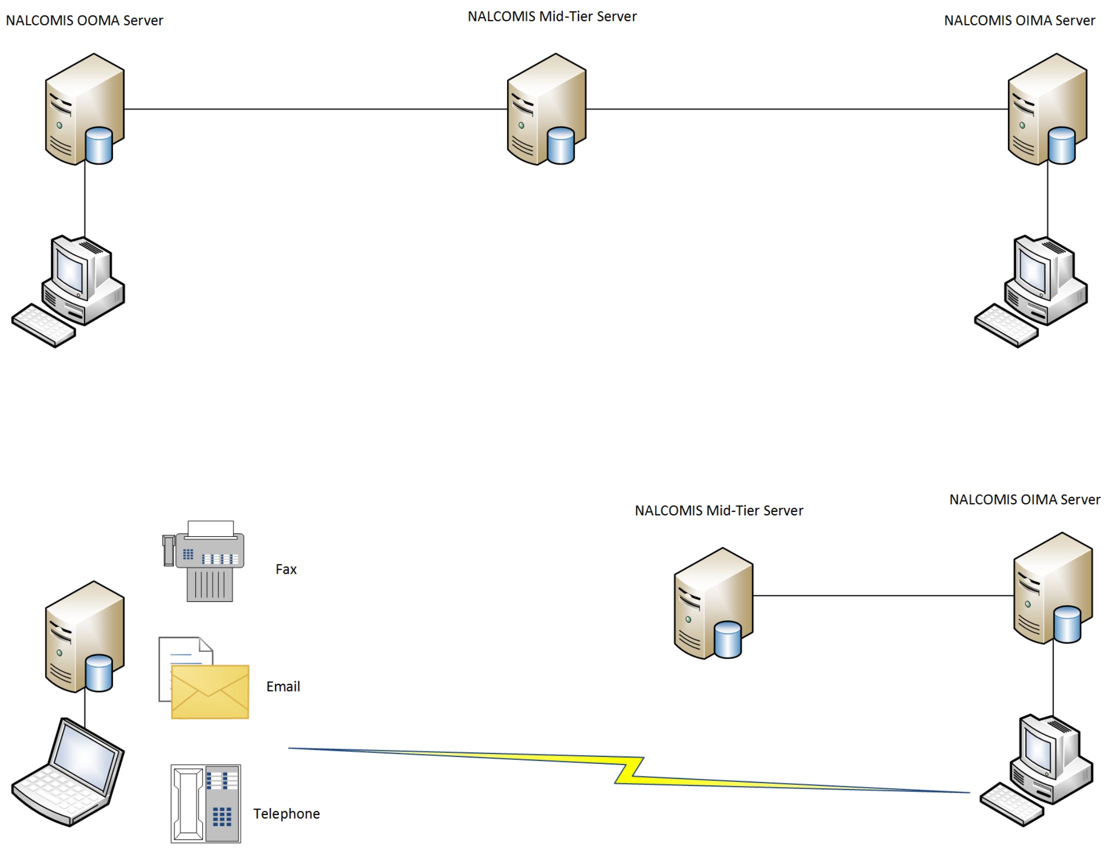


Figure 2.5. Current NALCOMIS Set Up at Home Station (top) and Deployed (bottom)

OOMA

At the O-level, NALCOMIS OOMA enables squadrons to manage maintenance and supply processes related to aircraft logs and records, material requisitions, and perform data analysis shared through a common database. As stated in the AVLOG MCWP, these processes support:

- Aircraft, engines, expeditionary airfield (EAF) components, and support equipment repair.
- Materiel requisitions.
- Direct and indirect support materiel control.
- Personnel, aircraft, and equipment assignment and deployment.
- Subcustody of equipment.
- Utilization of resources [4].

The common database avoids redundancy of functions and data and helps to provide better communication and response time between multiple databases [4].

OIMA

NALCOMIS OIMA provides similar functionality to the I-level, while also providing warehouse inventory and management tools. Internal communication at the I-level is established through online mailboxes and hard copy reports [4].

The O-level and I-level NALCOMIS servers communicate via an OOMA mid-tier server to pass material requisitions and status updates back and forth with each other (Figure 2.6).

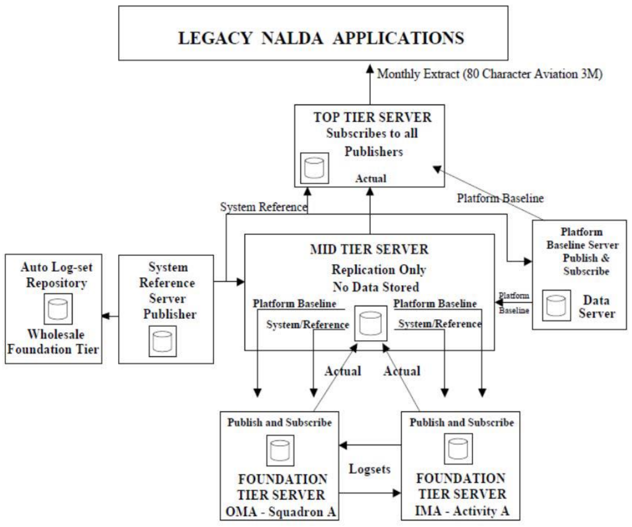


Figure 2.6. NALCOMIS Architecture. Source: [7].

As mentioned previously, this data is passed seamlessly and in real time between the O-level and I-level when the squadron's NALCOMIS OOMA is connected to the MALS level server. When flying squadrons deploy without their supporting MALS, those server connections are lost and data must be transferred via alternate means.

2.3 Deployed Operations

Flying squadrons will often deploy from their home stations to conduct training operations in various locations around the world. The O-level is able to take its NALCOMIS OOMA server on the deployment and continue to track necessary data and material requirements. As mentioned above, deploying severs the connection between the O-level and I-level NALCOMIS. The MALS supply department will send a detachment of personnel with the squadron along with a small pack-up of replacement parts to support the squadron. Obviously, not every required aircraft part can be brought on the deployment, and those requirements will need to be placed on order and tracked via alternate means.

In most cases, personnel from the squadron maintenance control section will provide a hand-written material requisition form to the supply detachment to fulfill. If the requested item is not available in the provided pack-up, the requisition is passed back to the home MALS via email, fax, or phone. The requisition is then manually entered into NALCOMIS for processing.

This method can lead to data processing errors, which may result in inventory discrepancies worth millions of dollars. Illegible handwriting, incorrect keystrokes, mishearing a phone conversation, etc., are all possible outcomes the more times a requisition is handled. If the connection between the O-level and I-level can be moved to the Internet, the direct connection between the two levels can be restored and reduce the number of opportunities to introduce errors.

CHAPTER 3: Enabling Technology

Transitioning a client-based NALCOMIS to the web requires ensuring that all necessary platforms interface together and result in no loss of data. In its simplest form, a web server, database, and Internet connectivity would be all that is required. In this chapter we will examine the components of an ideal, secure website and how they interface with each other.

3.1 Web Servers

The key component in a web-based NALCOMIS is a web server. A web server processes Hypertext Transfer Protocol (HTTP) requests, delivering web pages and their content to clients via a web browser (i.e., Microsoft Internet Explorer, Google Chrome, Mozilla Firefox). A web server can refer to hardware, software, or a combination of the two [8].

For hardware, a web server is a machine that hosts the relevant files for a website (e.g., hypertext markup language (HTML) files, cascading style sheets (CSS), pictures, JavaScript files) and sends the files to the requesting client. The server is connected to and accessed through the Internet via a domain name [8].

From the software perspective, web servers control how clients access files, typically through an HTTP server. An HTTP server is software that processes HTTP requests [8]. The software can execute scripts and be configured to manage permissions and access controls. Supporting different scripting languages enables web servers to deliver dynamic content, such as databases that are constantly being updated; essential to the concept of web based NALCOMIS. Examples of widely used web server products are Apache Tomcat (used in this proof of concept) [9] and Microsoft Internet Information Services (IIS) [10], [11].

Simply put, when a user wants to visit a web page, a browser sends a request over HTTP to retrieve the relevant files hosted on a web server. When the appropriate web server (hardware) receives the request, the requested file is sent back over HTTP via the HTTP server (software) (Figure 3.1) [8].

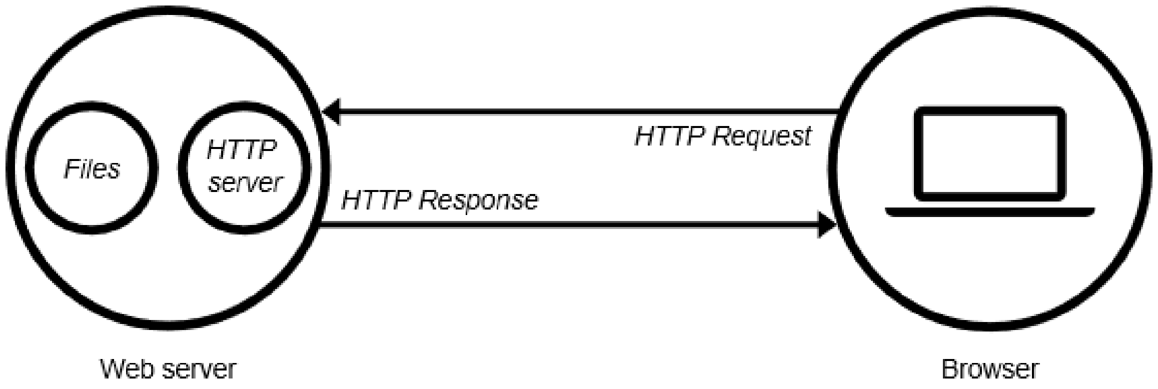


Figure 3.1. HTTP Request. Source: [8].

3.1.1 HTTP

As mentioned above, web servers process HTTP requests to deliver data over the Internet. HTTP is a textual, stateless protocol that functions at the application layer; meaning that it is readable by humans (textual), and that the server and client do not remember previous communications (stateless). The stateless aspect of HTTP means that items such as passwords or information typed into forms are not saved between sessions. In order to accomplish this an application server is needed [8].

HTTP also specifies how clients and servers communicate with one another. From the Mozilla website, "What is a web server?":

- Only clients can make HTTP requests, and then only to servers. Servers can only respond to a client's HTTP request.
- When requesting a file via HTTP, clients must provide the file's uniform resource locator (URL).
- The web server must answer every HTTP request, at least with an error message [8].

3.1.2 N-Tier Architecture

N-tier architecture refers to the client server model that separates presentation (what the user sees), application (how the program runs), and data (how data is manipulated) into a number of "n" layers. "N" can be any number, but two and three are the most prevalent, with three being the preferred method for most applications.

Two-Tier Architecture

Legacy applications connect to databases via a two-tier architecture, where the application (i.e., a Java program) connects directly via network protocols and ports to a database server. The client has direct access to the database, but only one request can be handled at a time. The server must finish handling the request before it can move on. This setup limits the mobility of the end user and ties the application to a local network [12] (Figure 3.2).

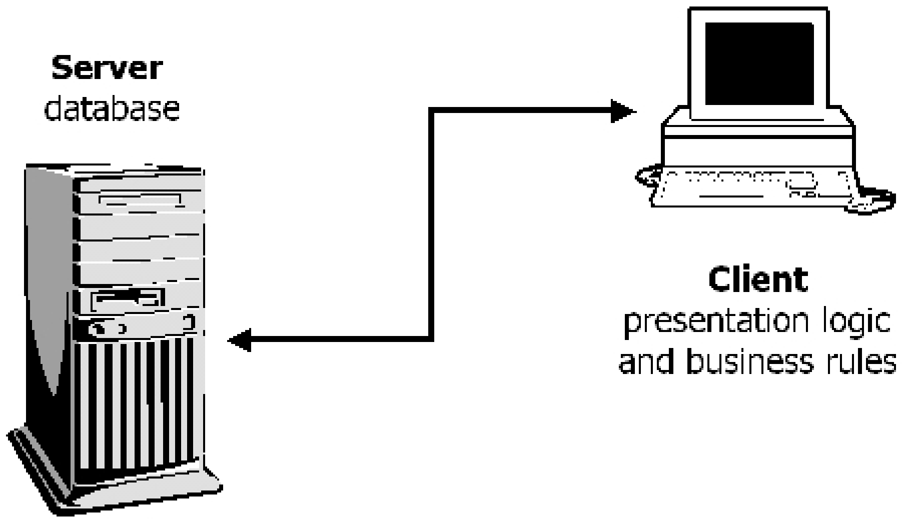


Figure 3.2. Two-Tier Architecture. Source: [13].

Three-Tier Architecture

In order to access data while mobile, the architecture must be upgraded to a three-tier setup. The new addition in this configuration is a middle-tier, such as a web server, between the database server and the end user, allowing for applications to be accessed via HTTP (Figure 3.3).

The introduction of the middle tier allows multiple users to access the database simultaneously, and makes the database available from anywhere. Clients are not affected by network and database operations; and network bottlenecks are reduced as the application layer only transmits the minimum required amount of data to execute tasks [14].

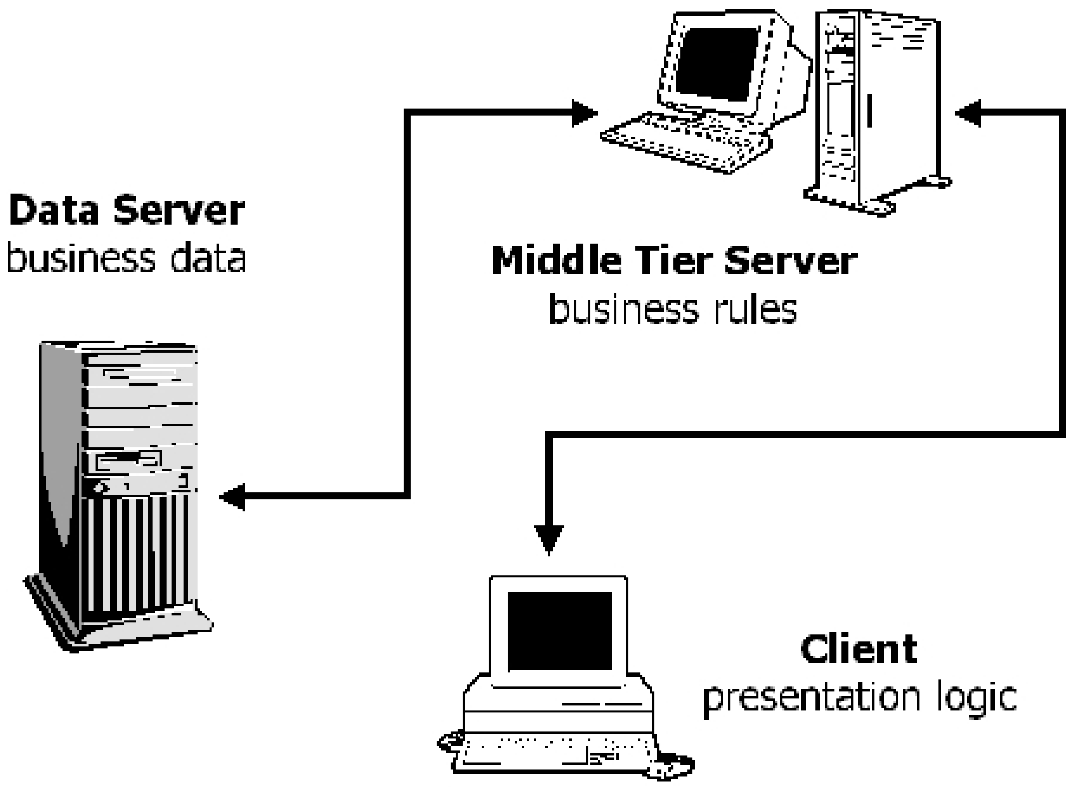


Figure 3.3. Three-Tier Architecture. Source: [13].

3.2 Relational Database

A database provides a way to store and retrieve information, typically utilizing tables with columns and rows to present the information. The rows in a table are objects of the same type and the data is usually related around a common concept, giving rise to the term relational database. The information in the database can be manipulated with a database management software (DBMS), and with a relational database a relational database management software (RDBMS) is used [15].

The data in a RDBMS is managed using a special language known as Structured Query

Language (SQL). SQL is a versatile language that allows for data insert, query, update, delete, schema creation and modification, and data access control [16].

3.2.1 Integrity Rules

Oracle's online Java Database Connectivity (JDBC) tutorial provides information on relational databases and how they function correctly. In order for database information to maintain accuracy and accessibility, the tables must follow "integrity rules" [15].

The first rule is row distinction. Duplication of row information will create conflicts in determining which is the appropriate row to select. Most DBMSs will allow a user to choose that duplicate rows not be allowed [15]. A second rule says that columns will not contain data presented as "repeating groups or arrays." [15] The third integrity rule looks at null values. A null value indicates that data is missing. A null value is not the same as a blank or zero. Blanks are equal to other blanks, and the same is true of zeroes; "but two null values are not considered equal." [15]

By following the first rule, row distinction, we can use one or more columns to identify a certain row. This column or columns becomes a primary key. If a column is a part of a primary key, it cannot be null, otherwise it would result in an incomplete identifier. This non-null rule is called entity integrity [15]. In Table 3.1, the National Item Identification Number (NIIN) column would serve as a primary key, as all parts in the NALCOMIS system have a NIIN assigned to them. Users can then run queries on the data with the key serving as a unique identifier.

Table 3.1. Example Table (not actual NALCOMIS data)

NIIN	**Nomenclature**	**COG**	**Project Code**	**Priority**	**Unit Price**	**Unit of Issue**
017248512	Radar	7R	706	02	750000	EA
996212358	Screw	9B	706	02	78	HD
012489965	Panel Rivet	9B	707	02	2.38	EA
004876348	Transistor	9B	AK1	05	36.12	DZ

3.3 Java Database Connectivity

The Oracle website provides online tutorials for many of its products, one of which, JDBC, is an important component of this research. "JDBC is an application program interface (API) that can access any kind of tabular data, especially data stored in a relational database" [15]. JDBC allows for database-independent connectivity between software applications written in the Java programming language and a wide range of databases. Simply put, JDBC acts as the middle-man between a software application and the database server [15].

3.3.1 Components

The tutorial goes on to describe the four components of JDBC:

- The JDBC API: The JDBC API provides programmatic access to relational data from the Java programming language. Using the JDBC API, applications can execute SQL statements, retrieve results, and propagate changes back to an underlying data source. The JDBC API can also interact with multiple data sources in a distributed, heterogeneous environment.

- JDBC Driver Manager: The JDBC DriverManager class defines objects which can connect Java applications to a JDBC driver. DriverManager has traditionally been the backbone of the JDBC architecture. It is quite small and simple. The Standard Extension packages javax.naming and javax.sql let you use a DataSource object registered with a Java Naming and Directory Interface (JNDI) naming service to establish a connection with a data source. You can use either connecting mechanism, but using a DataSource object is recommended whenever possible.

- JDBC Test Suite: The JDBC driver test suite helps you to determine that JDBC drivers will run your program. These tests are not comprehensive or exhaustive, but they do exercise many of the important features in the JDBC API.

- JDBC-ODBC Bridge: The Java Software bridge provides JDBC access via open database connectivity (ODBC) drivers. Note that you need to load ODBC binary code onto each client machine that uses this driver. As a result, the ODBC driver

> is most appropriate on a corporate network where client installations are not a major problem, or for application server code written in Java in a three-tier architecture [15].

The API and driver manager are used to connect to a database and then use a Java program and SQL commands to interact with a relational database. The test suite and bridge are used "to test web applications, or to communicate with ODBC-aware DBMSs" [15].

3.3.2 Two-Tier JDBC

Java applications run SQL commands to facilitate communication between an application and a database. This is handled by a JDBC driver that is loaded by a Java program and used to send SQL to the database and receive data back. Java provides a library called java.sql that lets database programmers work with databases and applications.

Each database vendor provides a JDBC Java archive (JAR) file to provide connectivity to the respective DBMS. Programs access the driver, located on the local machine, via the system CLASSPATH variable (tells Java where to look for user defined packages).

3.3.3 Three-Tier JDBC

In a three-tier format, a user provides information via a web page (written in HTML) and sent via HTTP GET (requesting data from a specified source) or POST (submitting data to be processed to specified source) methods. The data is then sent via the web server to an application server where JDBC begins to interpret the request and transmit it to the DBMS (Figure 3.4).

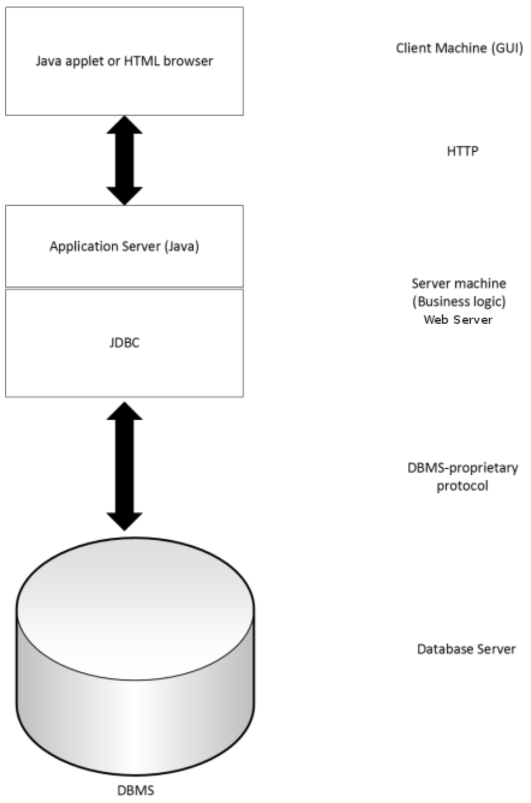


Figure 3.4. JDBC Architecture. Adapted from [17].

SQL queries sent from Java to the database return a ResultSet that is then translated back to HTML for viewing on the end user's web page. In this three-tier case, the JDBC driver resides in the web server's library folder instead of a local machine. Java code is compiled for the web and deployed to the web server as a web archive (WAR) file.

3.4 Security

Data sent over HTTP is not secure and accessing applications from the web introduces security risks; thus appropriate access controls and secure protocols like Hypertext Transfer Protocol Secure (HTTPS) must be implemented in order to reduce the likelihood of data theft and malicious attacks. Mitchell's thesis work [18] on security concerns highlights many of the areas that DOD websites must consider in order to protect their information. These include access control methods, public key infrastructure (PKI), and the use of the common access card (CAC).

3.4.1 Access Control Methods

"Access control includes the process of identification, authentication, and authorization for access to DOD protected information" [18]. Physical, logical, and administrative controls are implemented to ensure that information is protected from users who do not have the appropriate access levels. Physical controls may be actual physical distance, or, for example, a device like a key coded door. Logical control involves the use of both hardware and software tools to allow/deny access. These may include the use of "username and password; certificate-based authentication; and/or firewalls. Administrative controls are policies, procedures, and regulations that are in place to enforce the physical and logical access controls" [18].

Identification Credentials

In order to access a protected device and/or information, a user must validate his/her identity. This can be accomplished through the use of credentials which prove the identity of the user. Different credentials can provide different levels of authorization depending on a particular user and/or device [18].

Credentials are provided to a user or device through identity proofing, which involves the following steps:

- Validating the claimed identity of the user or device.
- Identifier naming and registering
- Generation of an authentication credential. This can be in the form of a PIN, PKI or other identification means.
- Binding the identity to an intended authentication method.
  The credential must be validated through the authentication process as part of the steps to gaining access [18].

Authentication

Authentication is the process of proving the identity of a user or device credential. Authentication is confirmed by at least one or a combination of the following three factors:

- Something the user knows (e.g., a password)
- Something the user has (e.g., CAC)
- Something the user is (e.g., fingerprint) [18]

Depending on the number of factors used determines the level of authentication. Using one of the above factors provides single-factor authentication; any two will provide two-factor; and all three results in three-factor authentication. The level of authentication is important to consider in the design of a device or how to protect information, and is dependent on available technology (e.g., fingerprint readers or iris scanners). Obviously the use of all three will "achieve the highest level of assurance" [18].

Authorization

Authentication alone does not provide unfettered access to DOD information. A user also requires appropriate authorization to access information. The authorization process determines if a user can access a system or not. After a user confirms his/her identity, typically through the use of a CAC and the PKI certificates stored on the card, access control lists can be used to validate the user's level of authorization.

Public Key Infrastructure

As Mitchell states, PKI provides two-factor authentication with something the user has (e.g., a hardware token) and knows (e.g., a PIN). "The infrastructure binds the public keys and user identity by the certificate authority (CA), and the validation authority (VA) ensures that the user's identity is unique within the CA domain. The registration authority ensures that the user is bound with its public key in a way that there is non-repudiation" [18]. PKI certificates offer the following services:

- Identification and authentication through digital signature of an identity challenge (i.e., on a website or an email program)
- Data integrity through digital signature of the information
- Confidentially through encryption
- Assists with technical non-repudiation through digital signatures [18]

PKI combines an X.509 formatted public key with a user's private key "to produce a unique authentication. The data element includes: name of user or device; start and end date of validation; public key; name of issuing CA; and the digital signature of the CA" [18]. The private key, something you have, is protected by requiring the user to input a PIN or password, something you know, again providing two-factor authentication [18]. Accessing non-public DOD information systems requires a DOD-approved PKI certificate. The PKI alone does not provide system access, though; proper authorization is still required [18].

3.4.2 Common Access Card

The DOD CAC website describes the CAC as a multi-purpose "smart card" that acts as an identification card for DOD personnel; stores information on its integrated circuit chip (ICC); and provides access to both physical spaces and computer networks and systems [19]. The CAC is the DOD approved hardware token, combining both identification and access control. "It is the merging of the user's personal identification with the PKI certificate and keys" [18].

The ICC stores the user's PKI certificates that allow the user to authenticate his/her identity to a DOD website. When a user attempts to access a PKI enabled website, the site will request a certificate which is verified with a user's personal identification number (PIN) [20].

Once the user is authenticated, the website is accessible over HTTPS allowing for encrypted transfer of data.

3.4.3 HTTPS

Using HTTPS, a browser accepts a public key certificate from a web server to verify the web page is hosted by that server. The browser will encrypt all plain text data with high-bit encryption, typically 128-bits. The browser encrypts with the server's public key and the server decrypts with its private key (Figure 3.5) [21].

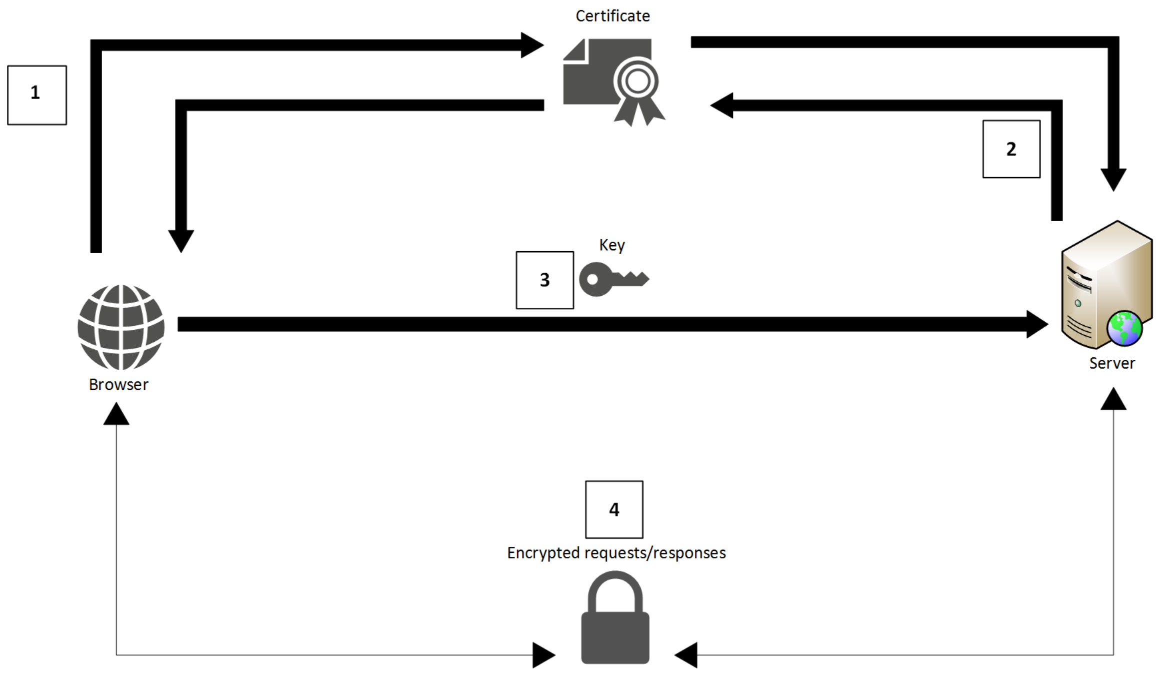


Figure 3.5. HTTPS Encryption. Source: [21].

3.5 Disconnected Operations

Oftentimes, network connectivity is lost which prevents the transmission of data. In these situations, many activities will resort to paper methods of data recording and the data entered into the database when connectivity is restored. Alternatively, activities could utilize hand-held devices loaded with database software, JDBC, and the appropriate application to continue storing data electronically, and then executing a synchronization process with a web server when the network is restored.

This process operates in the same way as the three-tier architecture described above, but allows for local storage on a mobile device in the event a network goes down (Figure 3.6).

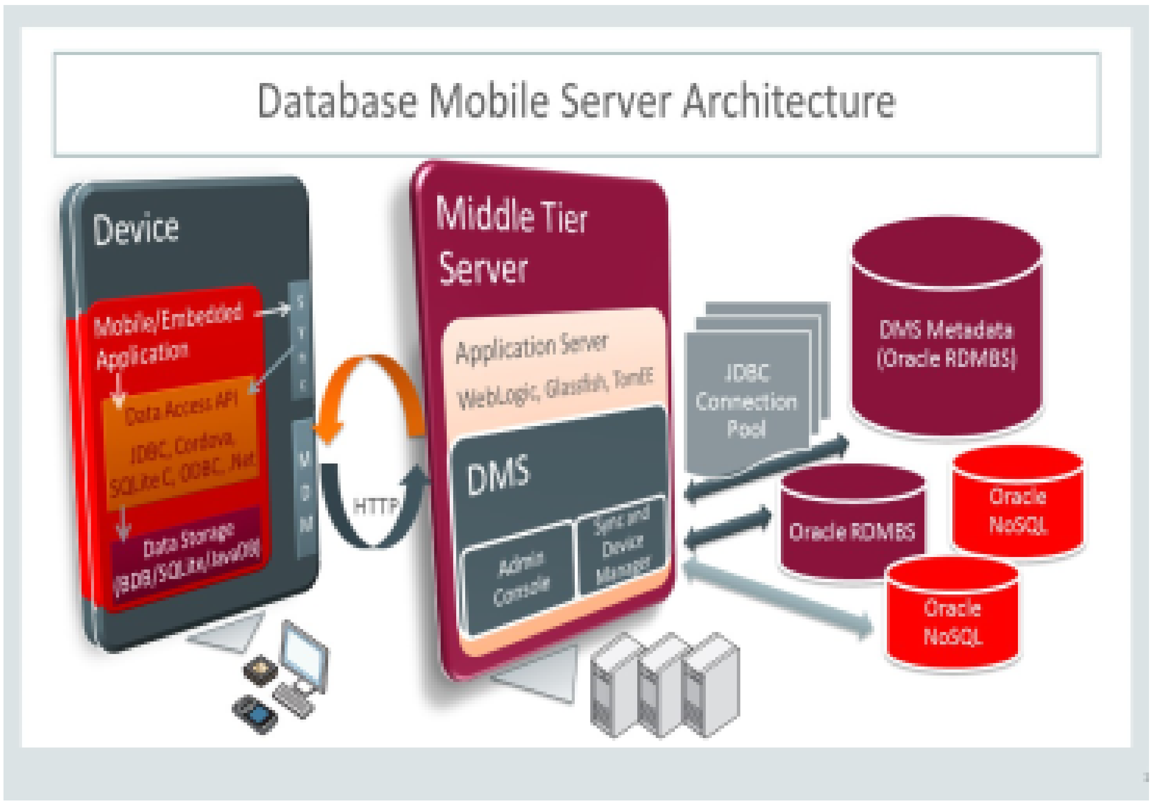

Database Mobile Server Architecture
Device
Mobile/Embedded Application
Data Access API
JDBC, Cordova, SQLite C, ODBC, .Net
Data Storage (BDB/SQLite/JavaDB)
Sync
MDM
HTTP
Middle Tier Server
Application Server
WebLogic, Glassfish, TomEE
DMS
Admin Console
Sync and Device Manager
JDBC Connection Pool
DMS Metadata (Oracle RDMBS)
Oracle RDMBS
Oracle NoSQL
Oracle NoSQL

Figure 3.6. Mobile Server Architecture. Source: [22].

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4: Architectural Implementation

4.1 Proposed Solution

A proposed solution to disconnected operations is to place the functionality of NALCOMIS on the web and have its functions accessible over the Internet. This will be accomplished utilizing the technologies and concepts presented in chapter three. Starting with an HTML5 form front-end (tier 1, presentation logic), data is transported over HTTPS through a web server (tier 2, business logic), and storing the data in a database (tier 3, database logic). The Java servlet running in the application server (Tomcat) takes the data sent from the HTML5 form (via HTTPS) and creates an SQL query which is sent via JDBC to the database. The database then sends the results back to the Java servlet following which the servlet then makes an HTML5 document and sends it back to the browser. Users would be authenticated via CAC login and their PKI certificates. Requisitions would be placed utilizing a simple web interface accessed via a CAC; data transmitted via HTML5 over HTTPS; JDBC converts the HTML5 data to SQL statements which are used to store the data on a NALCOMIS database server (Figure 4.1). Once the data hits the NALCOMIS server, the supply chain process takes over and the requisition is filled.

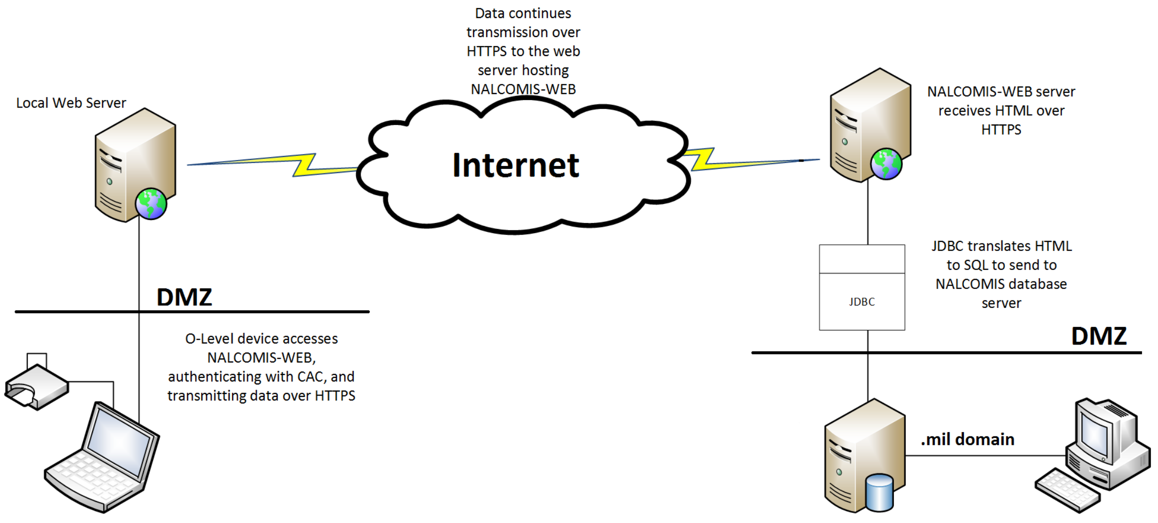

Data continues transmission over HTTPS to the web server hosting NALCOMIS-WEB
Local Web Server
Internet
NALCOMIS-WEB server receives HTML over HTTPS
JDBC translates HTML to SQL to send to NALCOMIS database server
JDBC
DMZ
O-Level device accesses NALCOMIS-WEB, authenticating with CAC, and transmitting data over HTTPS
DMZ
.mil domain

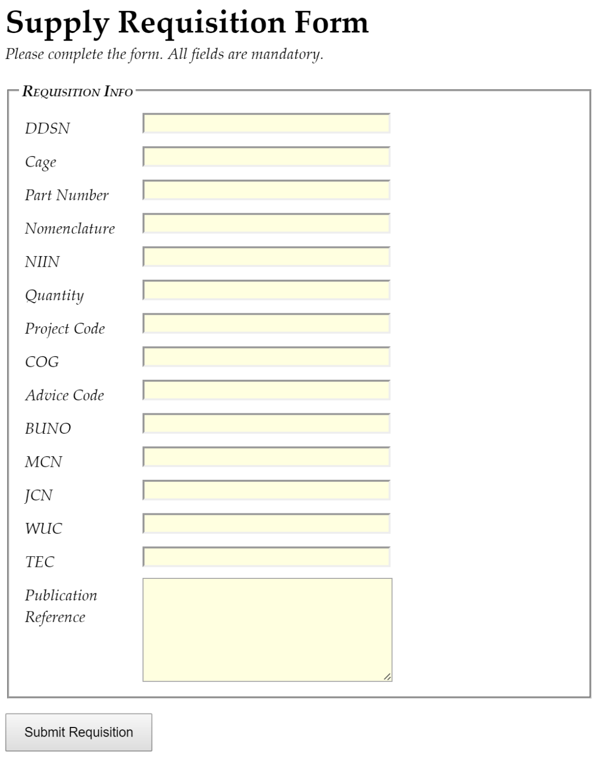
Figure 4.1. Proposed NALCOMIS-WEB

4.1.1 Proof of Concept

A proof of concept was developed to show the viability and utility of a NALCOMIS-WEB setup. Utilizing commercially available tools: Java Development Kit (JDK), SQL Developer, and Apache Tomcat; a simple model of NALCOMIS-WEB was built on a personal laptop computer (Dell XPS 9530 running the Windows 10 operating system). HTTP instead of HTTPS was used for data transportation because test of CAC access functionality was outside the scope of the initial proof of concept.

Data Entry Checking Using HTML5 & JavaScript

A simple web form (Figure 4.2) was created to capture the basic inputs required when a flying squadron's maintenance control submits a requisition to supply personnel while deployed from home station. The web form utilizes HTML5, JavaScript, and CSS.



Supply Requisition Form

Please complete the form. All fields are mandatory.

Requisition Info

DDSN	
Cage	
Part Number	
Nomenclature	
NIIN	
Quantity	
Project Code	
COG	
Advice Code	
BUNO	
MCN	
JCN	
WUC	
TEC	
Publication Reference	

Figure 4.2. Web Form for NALCOMIS-WEB

The form takes advantage of HTML5's error checking features, validating inputs and ensuring that all fields have been filled out by giving each field a "required" tag.

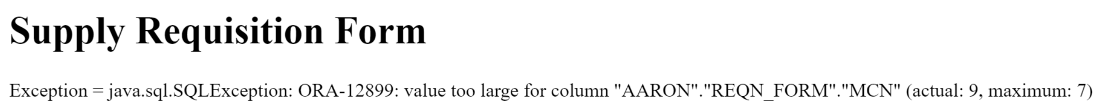
```
 <label for="ddsn">DDSN </label>
 <input name="ddsn" required><br>
```

The last block on the form is for a user to enter a publication reference. The form is set up to accept a minimum of 10 characters. The form will check that this requirement is met by calling a function, "validateComments()" [23] during data input.

```
<label for="reference">Publication Reference </label>
<textarea name="reference" oninput="validateComments(this)"
    required></textarea>
```

```
// from HTML5: The Missing Manual, 2nd Edition [23]
function validateComments(input) {
   if (input.value.length < 10) {
     input.setCustomValidity("You need to comment in more detail.");
   }
   else {
     // There's no error. Clear any error message.
     input.setCustomValidity("");
   }
} [23]
```

In order to maintain data integrity, the user input needs to be checked before sending it over the Internet to the web server. When the user clicks the "Submit Requistion" button, the HTML5 calls a function "validateForm()" [23] to confirm all form entries have been entered correctly. If a field is left blank and the user tries to submit, a warning box will pop up reminding the user to fill out the field. Submission of the form will fail if all fields have not been filled out correctly. This check also confirms that the user's web browser supports HTML5, as some older browsers may not support all of HTML5's features. If an error occurs, the user will be taken to a page that details what the error was (Figure 4.3). The user can then go back and correct the error and attempt to submit again. Only when the HTML5 client side check results in no errors will the form data be sent.



Supply Requisition Form

Exception = java.sql.SQLException: ORA-12899: value too large for column "AARON"."REQN_FORM"."MCN" (actual: 9, maximum: 7)

Figure 4.3. Error Found during Submission

```
// from HTML5: The Missing Manual, 2nd Edition [23]
function validateForm() {
  if (!Modernizr.input.required) {
    // The required attribute is not supported, so you need to check the
    // required fields yourself.

    // First, get an array that holds all the elements.
    var inputElements = document.getElementById("reqnForm").elements;

    // Next, move through that array, checking eaching element.
    for(var i = 0; i < inputElements.length; i++) {

      // Check if this element is required.
      if (inputElements[i].hasAttribute("required")) {
        // If this elemnent is required, check if it has a value.
        // If not, the form fails validation, and this function returns
            false.
        if (inputElements[i].value == "") {
          alert("Custom required-field validation failed. The form will not
              be submitted.");
          return false;
        }
      }
    }
    // If you reach this point, everything worked out and the browser
    // can submit the form.
    return true;
  }
} [23]
```

Database Creation

A simple schema of the NALCOMIS database was created in Oracle Database 11g XE (Express Edition) using SQL and SQL Developer, matching the inputs of the web form. The format of the database is shown in Table 4.1. The field type is based on the input expected for each entry. The size of each field is set to the character limit of each entry. For example, the document date serial number (DDSN) will always be eight characters in length.

Table 4.1. NALCOMIS Database Schema

Name	Type	Size
DDSN	VARCHAR2	8
CAGE	VARCHAR2	5
PART_NUMBER	VARCHAR2	20
NOMENCLATURE	VARCHAR2	30
NIIN	NUMBER	9
QUANTITY	NUMBER	4
PROJECT_CODE	VARCHAR2	3
COG	VARCHAR2	2
ADVICE_CODE	VARCHAR2	2
BUNO	NUMBER	6
MCN	VARCHAR2	7
JCN	VARCHAR2	11
WUC	NUMBER	3
TEC	VARCHAR2	4
REFERENCES	VARCHAR2	30

The Table 4.1 is generated in SQL Developer through a simple SQL script listed below:

```sql
CREATE TABLE "AARON"."REQN_FORM"(
  "DDSN" VARCHAR2(8 BYTE),
  "CAGE" VARCHAR2(5 BYTE),
  "PART_NUMBER" VARCHAR2(20 BYTE),
  "NOMENCLATURE" VARCHAR2(30 BYTE),
  "NIIN" VARCHAR2(9 BYTE),
  "QUANTITY" NUMBER(4,0),
  "PROJECT_CODE" VARCHAR2(3 BYTE),
  "COG" VARCHAR2(2 BYTE),
  "ADVICE_CODE" VARCHAR2(2 BYTE),
  "BUNO" NUMBER(6,0),
  "MCN" VARCHAR2(7 BYTE),
  "JCN" VARCHAR2(11 BYTE),
  "WUC" NUMBER(3,0),
  "TEC" VARCHAR2(4 BYTE),
  "REFERENCE" VARCHAR2(30 BYTE)
  );
```

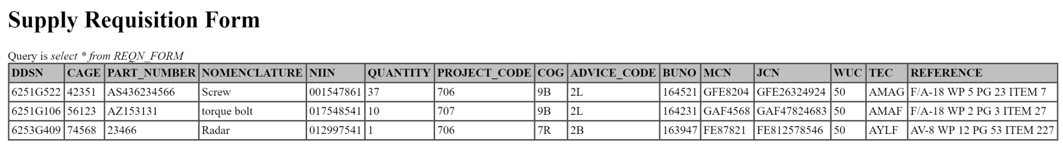
Java and JDBC

The process flow involves a web browser, the Tomcat web server, Java program and JDBC, and the SQL database. The Java program uses the JDBC driver to connect to the database and transmit the form data into the database. When the user clicks the "Submit Requistion" button, it establishes a connection to the database via the Tomcat web server and JDBC. In order for the web form and the database to connect to each other, a Java servlet that utilizes the JDBC driver is called in the web form at the time of submission. The HTML form interacts with the Java servlet and executes the code on submission. The data in the form fields is transferred over HTTP and stored in the database.

- Web Form interacting with Java servlet:

```
<form id="supplyRequisitionForm"
    action="../examples/servlets/servlet/reqnForm" onsubmit="return
    validateForm()" method="POST">
```

On successful submission, the data currently in the database will display in the user's browser (Figure 4.4).



Supply Requisition Form

Query is *select * from REQN_FORM*

DDSN	CAGE	PART_NUMBER	NOMENCLATURE	NIIN	QUANTITY	PROJECT_CODE	COG	ADVICE_CODE	BUNO	MCN	JCN	WUC	TEC	REFERENCE
6251G522	42351	AS436234566	Screw	001547861	37	706	9B	2L	164521	GFE8204	GFE26324924	50	AMAG	F/A-18 WP 5 PG 23 ITEM 7
6251G106	56123	AZ153131	torque bolt	017548541	10	707	9B	2L	164231	GAF4568	GAF47824683	50	AMAF	F/A-18 WP 2 PG 3 ITEM 27
6253G409	74568	23466	Radar	012997541	1	706	7R	2B	163947	FE87821	FE812578546	50	AYLF	AV-8 WP 12 PG 53 ITEM 227

Figure 4.4. Successful Form Submission

4.2 Marine Corps–Wide Application

After showing that a NALCOMIS-WEB is feasible, we next looked at increasing its scale. If accessing NALCOMIS over the web for use on deployments by a single squadron is possible, then the possibility of making it a full time web application utilized by the entire Marine Corps also exists. This section looks at the process of determining the hardware requirements for such a venture.

NALCOMIS processes transactions from both the supply and maintenance departments, but the focus of this research is on supply. The intent is to look at system load on a NALCOMIS server by the number of transactions that it processes over a given time. This data can then be used to determine the hardware requirements necessary to set up a Marine Corps wide NALCOMIS-WEB. All MALS could then be connected allowing for inventory data sharing and easier processing of lateral support requests between units.

In order to determine system load from the supply perspective, demand history needs to be examined. Twelve months of inventory management performance report (IMPR) data (February 2014–January 2015) was collected from the seven MALS in Marine Forces Pacific (MARFORPAC) (Table 4.2) for e xamination. The IMPR is a monthly report that measures different supply metrics, to include total monthly demands. Each MALS's demand data was placed into a Microsoft Excel spreadsheet and then used to calculate the number of demands per day, hour, minute, and second. For comparison, values were determined for monthly, quarterly, and the entire twelve month period rates. NALCOMIS data can be modeled as a Poisson distribution due to its bursty nature, allowing for easy manipulation and examination of the data.

Table 4.2. MARFORPAC Monthly Demand Data

	Jan	**Feb**	**Mar**	**Apr**	**May**	**June**	**July**	**Aug**	**Sept**	**Oct**	**Nov**	**Dec**
M-11	10446	7738	8229	8276	7887	10477	10282	9652	9565	10117	6508	9381
M-12	4483	2137	4058	3535	3370	2798	3179	3179	4172	4536	3269	5592
M-13	3875	3314	4849	4489	3773	3769	4658	4661	3894	3442	3119	3959
M-16	11233	7866	8186	8582	10209	9143	9607	8783	8913	10213	8372	10548
M-24	5435	4068	4970	5244	5251	5122	5186	5723	4749	4669	3572	4908
M-36	5683	3944	2956	3193	2449	4590	3780	4311	3729	4035	3351	6283
M-39	6958	6167	6408	6059	4416	6722	7528	6407	5633	6539	5210	6719
Total Reqn	*48113*	*35234*	*39656*	*39378*	*37355*	*42621*	*44220*	*42716*	*40655*	*43551*	*33401*	*47390*
Reqn/Day	1552.0323	1258.357	1279.2258	1312.6	1205	1420.7	1426.452	1377.9355	1355.1667	1404.871	1113.367	1528.7097
Reqn/Hour	64.668011	52.43155	53.301075	54.69167	50.20833	59.19583	59.43548	57.413978	56.465278	58.53629	46.39028	63.696237
Reqn/Min	1.0778002	0.873859	0.8883513	0.911528	0.836806	0.986597	0.990591	0.9568996	0.941088	0.9756048	0.773171	1.0616039
Reqn/Sec	0.0179633	0.014564	0.0148059	0.015192	0.013947	0.016443	0.01651	0.0159483	0.0156848	0.0162601	0.012886	0.0176934

After determining the arrival rates for different time periods, we need to figure out what the server requirements would be to transition NALCOMIS to a Marine Corps-wide web-centric application. This is best calculated by looking at server utilization at the unit level and then scaling up. Utilization can be calculated with Little's Law, represented by the formula:

$$\rho = \frac{\lambda}{\mu}$$

where ρ is server utilization, λ is arrival rate, and μ is service time. I used a hypothesized utilization rate of 20% from MALS-12 to calculate a hypothetical service rate. From these numbers I scaled up different scenarios to determine the necessary hardware requirements to run NALCOMIS on the web.

4.3 Results

The MALS-12 arrival λ of .001405 reqn/sec (Table 4.3) combined with the hypothetical ρ of 20% gives a μ of .007 reqn/sec. The μ value was carried forward for several scenarios to determine utilization and subsequently the number of servers required to support. At the MARFORPAC level (Table 4.4), ρ = 224%, requiring three servers running at 75% utilization. Doubling MARFORPAC's arrival rate to account for the rest of the Marine Corps brings the utilization to 449% (6 servers at 75%); and looking into the future as current platforms get older and newer ones become more complex, We scaled up the MARFORPAC arrival rate by five to see a utilization of 1121% (15 servers at 75%). We used 75% as an ideal utilization rate that makes efficient use of computing power and still leaves room for periods of increased server load.

Table 4.3. Squadron Yearly Rates

	Total Reqn	**Reqn/Day**	**Reqn/Hour**	**Reqn/Min**	**Reqn/Sec**
M-11	108558	297.41918	12.392466	0.2065411	0.003442
M-12	44308	121.39178	5.0579909	0.0842998	0.001405
M-13	47802	130.96438	5.4568493	0.0909475	0.001516
M-16	111655	305.90411	12.746005	0.2124334	0.003541
M-24	58897	161.36164	6.7234018	0.1120567	0.001868
M-36	48304	132.33973	5.5141553	0.0919026	0.001532
M-39	74766	204.83836	8.5349315	0.1422489	0.002371

Table 4.4. MARFORPAC Yearly Rate

Total Reqn	494290
Reqn/Day	1354.2192
Reqn/Hour	56.425799
Reqn/Min	0.94043
Reqn/Sec	0.0156738

The results show that with a minimum investment in hardware, a Marine Corps wide NALCOMIS-WEB could be implemented to create a fully interconnected MALS network. Maintenance departments could requisition parts much like the general public does on-line shopping on the Amazon website.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5: Future Work and Conclusions

5.1 Future Work

While a basic, functional proof of concept was created, future work still needs to be done in this area to refine and advance the project.

The actual utilization and/or service rates of the NALCOMIS system, to include maintenance transactions, need to be calculated in order to develop concrete rates and actual hardware requirements.

The proof of concept needs to be expanded to implement additional features such as CAC access and data transfer over HTTPS. The current version is only able to submit requisitions when devices are on the same network as the machine running the web server. Additional testing and setup is required to ensure that the NALCOMIS-WEB could be reached by necessary machines on any network. The development of a significantly upscaled version of the proof of concept could be handled by SPAWAR. Over time, the entire functionality of the current NALCOMIS could be added to NALCOMIS-WEB making a fully functional deployable system. A fully mobile system would allow maintenance personnel to use handheld devices to order replacement parts plane side during aircraft maintenance. Ideally, the system would be integrated and/or provide access to NAE systems to provide instant access to DLA and NAVSUP supply stock postures; as well as information provided by other MALS throughout the Marine Corps. It could lead to a truly real-time "one supply" system.

5.2 Conclusions

The deployable nature of Marine aviation and, in turn, aviation logistics, lends itself to utilizing mobile/web applications that are equally deployable. Transitioning NALCOMIS to a web-accessible environment is both feasible and can be done with relatively low financial investment.

The necessary functions are simple enough and appropriate software products readily available. If the focus is kept on the security and functionality, a simple web-based concept such as NALCOMIS-WEB can move the Marine Corps away from antiquated, and arguably more mistake-prone, methods. Aviation logistics requires fast, responsive service and the NALCOMIS-WEB concept can only improve current capabilities and practices.

List of References

[1] *Marine Corps Warfighting Publication 3-2: Aviation Operations*. Washington, D.C.: United States Marine Corps, 2000.

[2] GlobalSecurity. (2015, Apr.). Marine Expeditionary Force. [Online]. Available: http://www.globalsecurity.org/military/agency/usmc/mef.htm

[3] 3rd Marine Aircraft Wing. (2015, Apr.). [Online]. Available: http://en.wikipedia.org/wiki/3rd_Marine_Aircraft_Wing

[4] *Marine Corps Warfighting Publication 3-21.2: Aviation Logistics*. Washington, D.C.: United States Marine Corps, 2012.

[5] GlobalSecurity. (2015, Apr.). Aviation Combat Element Logistics Organization. [Online]. Available: http://www.globalsecurity.org/military/library/policy/army/fm/90-31/fig9-42.gif

[6] Defense Logistics Agency purchasing activities. (2015, Apr.). [Online]. Available: http://www.wingovernmentcontracts.com/defense-logistics-agency-purchaseing-activities.htm

[7] E. Henzler and M. Williams, "Is electronic life-cycle tracking of aircraft parts degrading readiness," Master's thesis, Naval Postgraduate School, Monterey, CA, 2013.

[8] What is a web server. (2015, Dec.). [Online]. Available: https://developer.mozilla.org/en-US/Learn/What_is_a_web_server#Communicating_through_HTTP

[9] Apache Tomcat. (2016, Aug.). [Online]. Available: https://tomcat.apache.org

[10] IIS. (2016, Aug.). [Online]. Available: http://www.iis.net

[11] June 2016 Web Server Survey. (2016, Aug.). [Online]. Available: http://news.netcraft.com/archives/2016/06/22/june-2016-web-server-survey.html

[12] Two-tiered client/server limitations. (2016, Jan.). [Online]. Available: http://www.n-tier.com/articles/cslimits.html

[13] Three-tier architecture. (2016, Aug.). [Online]. Available: http://www.linuxjournal.com/article/3508

[14] Three-tier client/server architectures. (2016, Jan.). [Online]. Available: http://www.n-tier.com/articles/csovervw.html

[15] The Java tutorials. (2015, May). [Online]. Available: https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html

[16] Tables, rows, and columns. (2016, Jan.). [Online]. Available: http://www-01.ibm.com/support/knowledgecenter/SSPK3V_6.3.0/com.ibm.swg.im.soliddb.sql.doc/doc/tables.rows.and.columns.html

[17] JDBC Overview. (2015, May). [Online]. Available: http://www.oracle.com/technetwork/java/overview-141217.html

[18] K. Mitchell, "Security concerns In accessing naval e-learning with personal mobile devices," Master's thesis, Naval Postgraduate School, Monterey, CA, 2014.

[19] Common Access Card (CAC). (2015, May). [Online]. Available: http://www.cac.mil/common-access-card/

[20] Common Access Card (CAC) Security. (2015, May). [Online]. Available: http://www.cac.mil/common-access-card/cac-security/

[21] S. Miner. (2015, May). CPS 353: Internet programming, sessions and security. [Online]. Available: http://www.math-cs.gordon.edu/courses/cps353/lectures/12_sessions_and_security.html

[22] Oracle database mobile server overview. (2015, May). [Online]. Available: http://www.oracle.com/technetwork/database/database-technologies/database-mobile-server/overview/index.html

[23] M. MacDonald, *HTML5: The Missing Manual, 2nd Edition*. New York: O'Reilly Media, 2013.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California